

# Бенчмаркинг для системы моделирования кристаллических решёток

Попов Владимир, Б01-411

## 1 Классы нагрузок и подходящие бенчмарки

В представленном ранее проекте требуется определить, какое оборудование установить на сервере и на лабораторных машинах. Это можно решить, изучив соответствующие бенчмарки.

Таблица 1: План бенчмаркинга инфраструктуры

Компонент	Бенчмарк	Цель
CPU (Вычисления)	SPECrate Floating Point Base	Оценка многопоточной производительности для вычислений с плавающей точкой.
GPU (Визуализация)	SPECviewperf	Гарантирует, что выбранные GPU справятся с рендерингом в PyQt5+OpenGL миллионов атомов без задержек
Storage	FIO, STREAM	Оценка скорости ввода/вывода локальных конфигураций на лабораторном железе. Оценка пропускной способности памяти и работы БД, так как данные будут подгружаться с удалённого сервера.

## 2 Решение проблем бенчмаркинга

Синтетические тесты могут не отражать реальную нагрузку молекулярной динамики, которая зависит от кэш-памяти при построении списков соседей и имеет специфический паттерн доступа к памяти.

SPECrate FP Включает реальные научные приложения, в том числе задачи вычислительной химии и физики, и SPECviewperf включает тесты визуализации научных данных.

STREAM использует массивы размером более чем в 4 раза превышающие размер кэша последнего уровня, что соотносится с нашим случаем, ведь подгружать данные хочется

максимально большими блоками, и параллелизация будет происходить на кубы непомещающиеся в кэш. При этом STREAM использует последовательный доступ к памяти, что соответствует обходу атомов в суперячейке, а в FIO можно протестировать случайный доступ, что соответствует выборке конфигураций из БД.

В итоге

- **SPECrate**: CPU, компилятор, библиотеки, ОС
- **SPECviewperf**: GPU, драйверы OpenGL, оконная система
- **FIO**: дисковая подсистема, СУБД, файловая система
- **STREAM**: CPU, контроллер памяти